Setting up MySQL on the Cluster after Rebuild

Samantha Wohlstadter

February 10, 2019

(In Progress)

1 Setting up the Configuraton File

When first starting MySQL, you need to make sure the correct paths to the socket file and the databases are specified in the configuration file. The configuration file is located on the CE at `/etc/my.cnf`.

The file `/opt/rocks/mysql/my.cnf` should contain the correct information which can simply be copied over to `/etc/my.cnf` since this file is not looked at by MySQL when it runs. If this file does not exist, the correct socket should be `socket=/var/opt/rocks/mysql/mysql.sock`, and the data directory should be `datadir=/var/opt/rocks/mysql/`.

Make sure to specify the paths under both the [mysqld] and the [client] headings in the configuration file. After making changes to the configuration file, make sure to restart apachectl and the mysql server.

```
/usr/sbin/apachectl graceful
```

```
/opt/rocks/mysql/support-files/mysql.server restart
```

2 Importing Databases

The databases used in mysql for the cluster tend to be in "InnoDB" format. What that means for importing databases is that one cannot simply copy the folders directly from the old databse to the new one. MySQL cannot read the files this way. Instead, the old databases must be 'dumped' using the mysqldump command.

2.1 The Proper Procedure

To export a database, run the following:

```
mysqldump -u root -p (database) > (database).sql
```

To import a database, first create the database in mysql using "`create database (name);`" Then, quit MySQL and run the following:

```
mysql -u root -p (database) < (database).sql
```

Once you have imported the databases, be sure to give the appropriate permissions.

```
GRANT ALL PRIVILEGES ON (database)* TO (user)@localhost
```

Be sure to update appropriate configuration files when changes are made, such as LocalSettings for the wiki page.

2.2 The Brute-Force Procedure

If you do not have the .sql files, and the old mysql server is not able to run, the databases can still be accessed. You will need the "ibdata1" file from the old mysql server to be able to read the data from the databases, however. The old mysql files are located on nas1 in `CEBackup-20180803/var/lib/mysql`.

The following procedure was performed on a different machine than the one hosting the new mysql server, though it can be done on the same machine.

1. Stop MySQL using `/opt/rocks/mysql/support-files/mysql.server stop`
2. Create a copy of the current data directory (in this case it is /var/opt/rocks/mysql) and rename it to create a backup.
3. Copy the desired database folders (excluding "mysql") from the old server to the new server.
4. Start MySQL using /opt/rocks/mysql/support-files/mysql.server start
5. Log in to mysql and run "`use (database); show tables; select * from (table);`" on each database that was just copied over. You should get "table does not exist". This is okay at this point, you are "showing" mysql that these databases and tables exist in its data directory now, even though it can't read them yet. This is an important step!
6. Quit mysql, then stop the service using the following command:

   ```
   /opt/rocks/mysql/support-files/mysql.server stop
   ```

7. Replace the ibdata1 file with the file of the same name from the old MySQL server.
8. (Optional?) Replace the "mysql" database with the databse of the same name from the old server. (You must know the password from the old server to log in to MySQL once this step is performed).
9. Start the mysql service with the following:

   ```
   /opt/rocks/mysql/support-files/mysql.server start
   ```

10. Log in to mysql and test access to the data in the database with "`use (database); show tables; select * from (tablename)`" You should get a table of data as the output.

11. If the above step was successful, log out of mysql and use mysqldump as described in Section 2.1 to create the .sql files and import them as above. If you used the same machine to perform this procedure, make sure to rename the current mysql directory to something else, and rename the backup to "mysql/" before importing the old databases into the new MySQL server.

You should then be able to access the data in the new server.

3 Errors

3.1 Can't Connect through Socket

This error occurs when MySQL can't find the mysql.sock file. The error will look similar to the following:

```
Error 2002: Can't connect to local MySQL server through socket
'var/lib/mysql/mysql.sock.'
```

The sock file is either missing or the path specified in `/etc/my.cnf` is incorrect. First, check to see if MySQL is running with the following:

```
netstat -tap | grep mysql
```

If nothing appears, try starting the MySQL server with:

```
/opt/rocks/mysql/bin/mysqld start
```

If MySQL is running, you need to find where the .sock file is, and specify the path to it in /etc/my.cnf.

You may also see an error similar to this in the error log:

```
/opt/rocks/mysql/bin/mysqld: Can't change dir to '/var/lib/mysql/'
(Errcode: 2 - No such file or directory)
```

The solution to this error is to change the path to the .sock file in `/etc/my.cnf`.

The following may be used instead, but is not recommended because it is long and tedious to type.

```
mysql --socket=/var/opt/rocks/mysql/mysql.sock -u (user) -p
```

3.2 InnoDB Can't open Table

This error may look similar to the following:

```
[Warning] InnoDB: Cannot open table wikidatabase/archive from the
internal data dictionary of InnoDB though the .frm file f or the
table exists.  See http://dev.mysql.com/doc/refman/5.6/en/innodb-
troubleshooting.html for how you can resolve the problem.
```

This occurs when MySQL can't read the tables in the databases. This most likely occurs when the files from an InnoDB format database are copied to a different MySQL server without the ibdata1 file. MySQL doesn't know how to read the `.frm` files contained in the database files without the ibdata1 file. To solve this issue, see Section 2.2.

3.3 Resetting the Password

If the root password is not known, it can be reset using the following method. If the password is changed, it will also have to be updated for everying that uses MySQL (which is a lot of things) so they can still access the databases needed to run.

1. Run `opt/rocks/mysql/bin/mysqld_safe --skip-grant-tables` in one terminal; it should continue to run while the password is reset. If the `/etc/my.cnf` file is not configured correctly, use the additional flags:

   –basedir=/opt/rocks/mysql –datadir=/var/opt/rocks/mysql

–log-error=/var/opt/rocks/mysql/uscms1.fltech-grid3.fit.edu.err

2. In a new terminal, run `mysql -u root` to log into MySQL.
3. Depending on the version of MySQL, this step differs slightly. In pre-5.7 MySQL versions (such as what the cluster currently uses), run the following:

   `UPDATE mysql.user SET Password=PASSWORD('password') WHERE User='root';` (Keep the single quotes.)

   If MySQL version 5.7 or above is used, replace "`Password=`" with "`authentication_string=`"
4. Run `FLUSH PRIVILEGES;` then quit mysql.
5. Run `mysqladmin -u root -p shutdown`. At this point, mysqld_safe in the first terminal should end.
6. Start MySQL with `/opt/rocks/mysql/support-files/mysql.server start`

(Note: The LocalSettings file for the wiki does not like having double quotes in a password, since it uses double quotes to surround the password string. It is not reccomended to use double quotes in the password of the MySQL user that will access the wikidatabase.)

3.3.1 Errors Starting mysqld_safe

If the correct paths for the socket file and the data directory are not specified in `/etc/my.cnf`, the following errors might occur:

```
190112 20:40:46 mysqld_safe Starting mysqld daemon with databases
from /var/opt/rocks/mysql 190112 20:40:46 mysqld_safe mysqld from
pid file /var/run/mariadb/ mariadb.pid ended
```

```
2019-01-12 20:55:30 23092 [ERROR] /opt/rocks/mysql/bin/mysqld:
Can't create/write to file '/var/run/mariadb/mariadb.pid' (Errcode:
2 - No such file or directory)  2019-01-12 20:55:30 23092 [ERROR]
Can't start server:  can't create PID file:  No such file or directory
```

Another error that may occur is “permission denied”. To fix this, check to make sure that the mysql data directory is owned by mysql. If it is not, run `chown -R mysql:mysql /var/opt/rocks/mysql/`